Exercițiul 5

Îmbunătățiri:

1. Adaugarea unui contor de acces pentru fiecare fir de execuție:
   ```
   private static AtomicIntegerArray accessCount = new AtomicIntegerArray(n);
   ```
2. Verificarea contorului înaintea avansarii la un nou nivel:

   ```
   if (k != i && level.get(k) >= L && (victim.get(L) == i || accessCount.get(i) > accessCount.get(k)))
   ```

   În plus față de condițiile inițiale, condiția “sau” dintre contorul de acces și victima curentă este folosită pentru a preveni situațiile în care trei sau mai multe thread-uri așteaptă să avanseze la același nivel, iar un thread mai recent (dar nu cel desemnat drept victimă) avansează înaintea primului thread care așteaptă, fără a se lua în considerare cât de frecvent a accesat acesta secțiunea critică. Această condiție asigură respectarea principiului de fairness, chiar și atunci când mai multe thread-uri concurează pentru a avansa la același nivel.

Fairness integrat – În acest mod, prin rotația bazată pe `accessCount`, implementăm fairness fără o metodă externă de verificare. Algoritmul permite accesul echitabil, deoarece fiecare thread trebuie să aștepte până când toate celelalte au accesat secțiunea critică de un număr aproximativ egal de ori.

Astfel, această modificare asigură 0-bounded waiting și fair access la secțiunea critică.